Volume

1

Installer CUDA sous Windows



Manuel d'installation

Table des matières



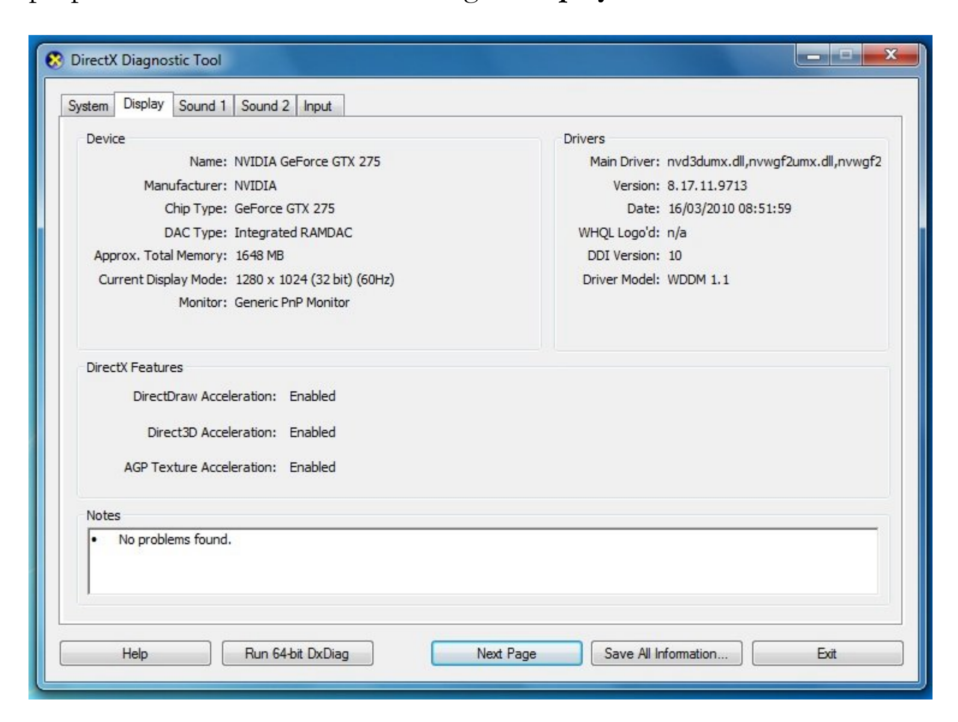
Installation de CUDA & configuration

Dans ce chapitre sont décrites les différentes étapes qui permettent de faire tourner un programme en CUDA sur un ordinateur équipé de Windows (32 ou 64 bits).

Pour installer CUDA, c'est très simple, il faut faire tout ce qui est marqué ci-dessous.

S'assurer que l'ordinateur supporte CUDA

Trouvez quel modèle de carte graphique est présent dans l'ordinateur en ouvrant une invite de commande et en tapant l'instruction **dxdiag** qui ouvre la fenêtre des propriétés DirectX. Sélectionnez l'onglet **Display**.



Allez ensuite sur la liste de NVIDIA des produits supportant CUDA et cherchez votre modèle. Toutes les cartes GeForce des séries 8000 ou au-dessus, qu'il s'agisse de PC, de portables ou de chipset ION, peuvent faire tourner du CUDA. Il en est de même pour les Tesla et les Quadro. Vérifiez dans la liste que votre modèle est tout de même bien présent.

Si votre ordinateur ne peut malheureusement pas faire tourner du CUDA, il reste toujours le mode émulation qui permet de compiler et de lancer du code, très lentement toutefois.

Installer le driver

Le driver est ce qui permet d'accéder aux dernières possibilités de la carte NVIDIA, y compris de l'aide pour CUDA.

Rendez vous sur la page de téléchargement de driver de NVIDIA, sélectionnez votre système d'exploitation ainsi que votre carte graphique et télécharger le driver le plus récent.



A la fin de l'installation de ce dernier, il est nécessaire de redémarrer l'ordinateur.

Installer le driver de développement, le Toolkit et le SDK

Le Toolkit CUDA permet de compiler des programmes CUDA tandis que le SDK contient des projets que l'on peut utiliser pour se familiariser avec CUDA ou en commencer un propre. Il faut se rendre sur la page de téléchargement de NVIDIA CUDA ZONE et sélectionner les programmes suivant le système d'exploitation.

Pour un ordinateur portable, la dernière version compatible de CUDA est la 2.2. Pour un ordinateur de bureau, il s'agit de la 2.3.

Windows

Developer Drivers for WinXP (197.13)	32-bit 64-bit	
Developer Drivers for WinVista & Win7 (197.13)	32-bit 64-bit	
Notebook Developer Drivers for WinXP	32-bit 64-bit	
Notebook Developer Drivers for WinVista & Win7	32-bit 64-bit	
CUDA Toolkit • C/C++ compiler • CUDA Visual Profiler • OpenCL Visual Profiler • GPU-accelerated BLAS library • GPU-accelerated FFT library • Additional tools and documentation	32-bit 64-bit	Getting Started Guide for Windows Release Notes CUDA C Programming Guide CUDA C Best Best Practices Guide OpenCL Programming Guide OpenCL Best Best Practices Guide OpenCL Implementation Notes CUDA Reference Manual API Reference PTX ISA 2.0 Visual Profiler User Guide Visual Profiler Release Notes Fermi Compatibility Guide Fermi Tuning Guide CUBLAS User Guide CUFFT User Guide License
NVIDIA Performance Primitives (NPP) library	32-bit 64-bit	
CULA: GPU-accelerated LAPACK libraries	download	more info
NVIDIA Parallel Nsight for Visual Studio		more info
CUDA Fortran from PGI	download	more info
GPU Computing SDK code samples	32-bit 64-bit	Release Notes for CUDA C Release Notes for DirectCompute Release Notes for OpenCL CUDA Occupancy Calculator License
NVIDIA OpenCL Extensions		Compiler_Options D3D9 Sharing D3D10 Sharing D3D11 Sharing Device Attribute Query Pragma Unroll

Essayer les exemples

Lancez les exemples du SDK afin de vous assurer que tout fonctionne correctement. Pour CUDA 2.2, cherchez le navigateur NVIDIA CUDA SDK dans le menu Démarrer. Pour CUDA 2.3, cherchez le navigateur NVIDIA GPU Computing SDK. Si les exemples fonctionnent, vous avez installé avec succès le bon driver CUDA.



Tester votre installation en compilant un exemple

Ouvrez le dossier CUDA SDK en ouvrant le navigateur du SDK et en sélectionnant Files dans n'importe lequel des exemples. Allez au dossier **src** (CUDA 2.3) ou **projects** (CUDA 2.2) et ensuite dans un exemple. Ouvrez le projet Visual Studio 9.0 dans Visual C++. Compilez-le en choisissant Build > Rebuild Solution.

Lancez-le via Debug > Start without Debugging. Si le projet compile sans erreur et tourne correctement, félicitations ! Vous avez en effet correctement installé tous les éléments dont vous aviez besoin pour développer en CUDA sous Windows.



Comment intégrer CUDA à Visual C++

Let's check how we can support CUDA compilation with VC++ !

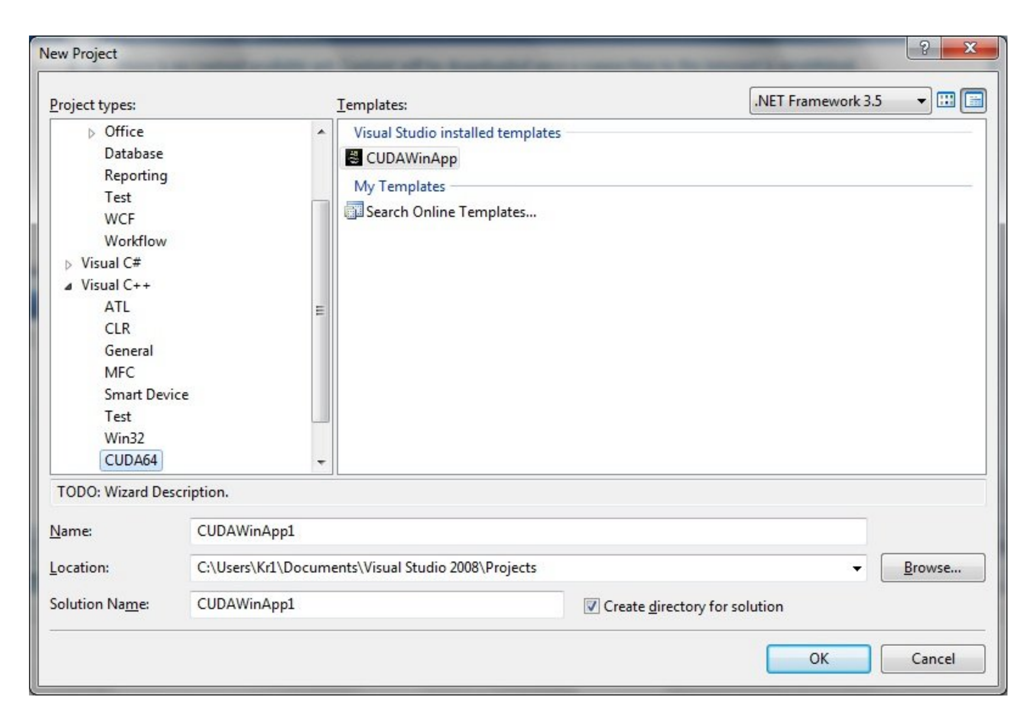
Dans le dossier dans lequel vous avez installé CUDA (le plus probablement C:\CUDA) se trouve nvcc.exe, le compilateur qui permet de générer du code assembleur PTX à partir d'un fichier .cu. Mais dans le monde de Windows, la plupart des programmeurs sont très satisfaits par l'IDE Visual Studio.

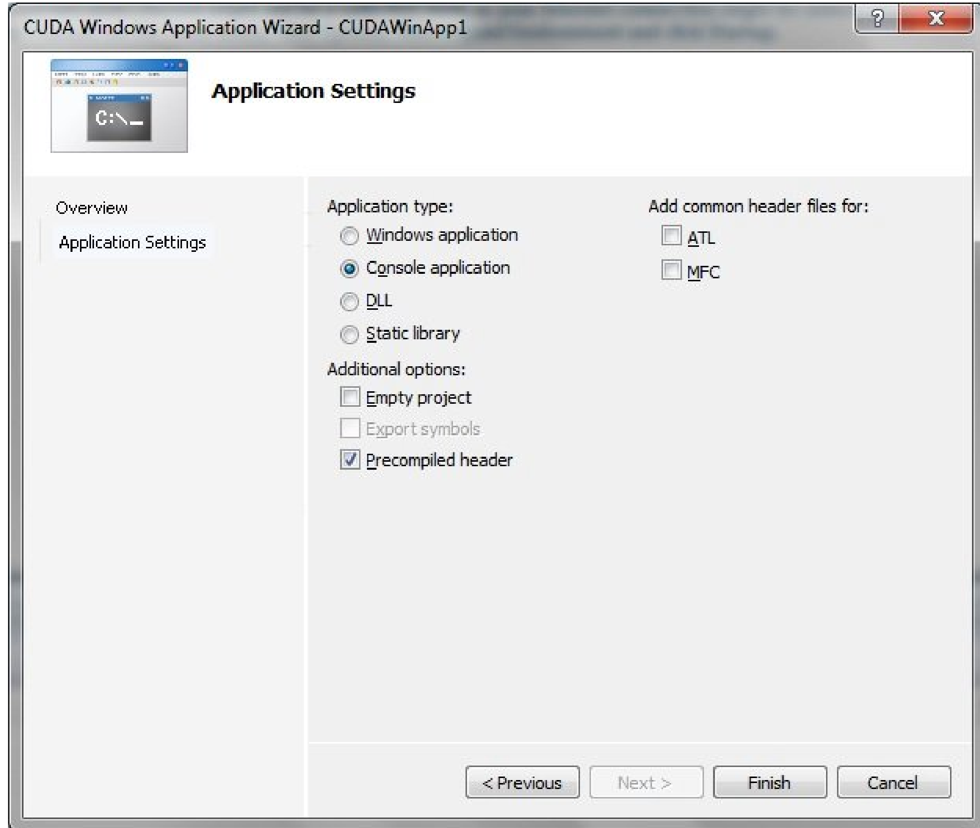
Le compilateur CUDA a des dépendances avec le compilateur C++. Il supporte indifféremment Visual C++ 7.1, 8.0 ou 9.

Méthode 1 – Installer CUDA VS Wizard

CUDA est prévu pour le compilateur de Visual Studio, sous Windows, alors que NVIDIA ne propose strictement aucun moyen d'intégrer le SDK dans l'IDE. C'est pourquoi Kaiyong Zhao a créé un modèle de projet pour cet IDE. Ce modèle est compatible avec Visual Studio 2005 (8.0) et 2008 (9.0).

Il vous suffit d'aller sur le site web du projet pour télécharger le modèle, puis de l'installer. Ainsi, un nouveau type de projet sera disponible à la création.


New Project
Project types:
Templates:
.NET Framework 3.5
Office
Database
Reporting
Test
WCF
Workflow
Visual C#
Visual C++
ATL
CLR
General
MFC
Smart Device
Test
Win32
CUDA64
Visual Studio installed templates
CUDAWinApp
My Templates
Search Online Templates...
TODO: Wizard Description.
Name:
CUDAWinApp1
Location:
C:\Users\Kr1\Documents\Visual Studio 2008\Projects
Browse...
Solution Name:
CUDAWinApp1
Create directory for solution
OK
Cancel


CUDA Windows Application Wizard - CUDAWinApp1
Application Settings
Overview
Application Settings
Application type:
Windows application
Console application
DLL
Static library
Additional options:
Empty project
Export symbols
Precompiled header
Add common header files for:
ATL
MFC
< Previous
Next >
Finish
Cancel

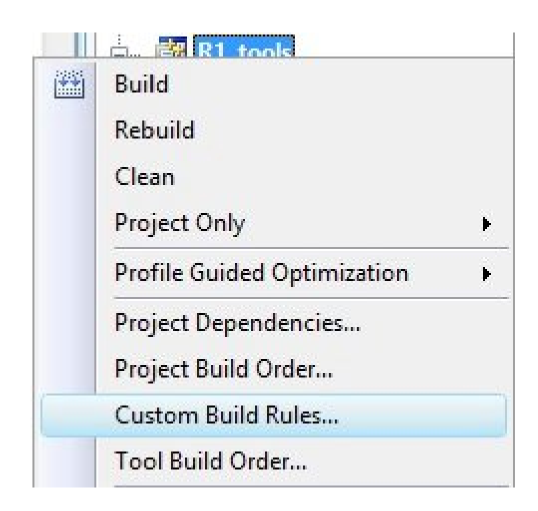
Méthode 2 – Construire son projet en partant du template du SDK

Le SDK contient un dossier nommé **Template** que l'on peut utiliser pour entamer un projet depuis le début. S'il s'agit d'intégrer du CUDA dans un projet déjà existant, l'exemple intitulé cppIntegration indique la marche à suivre.

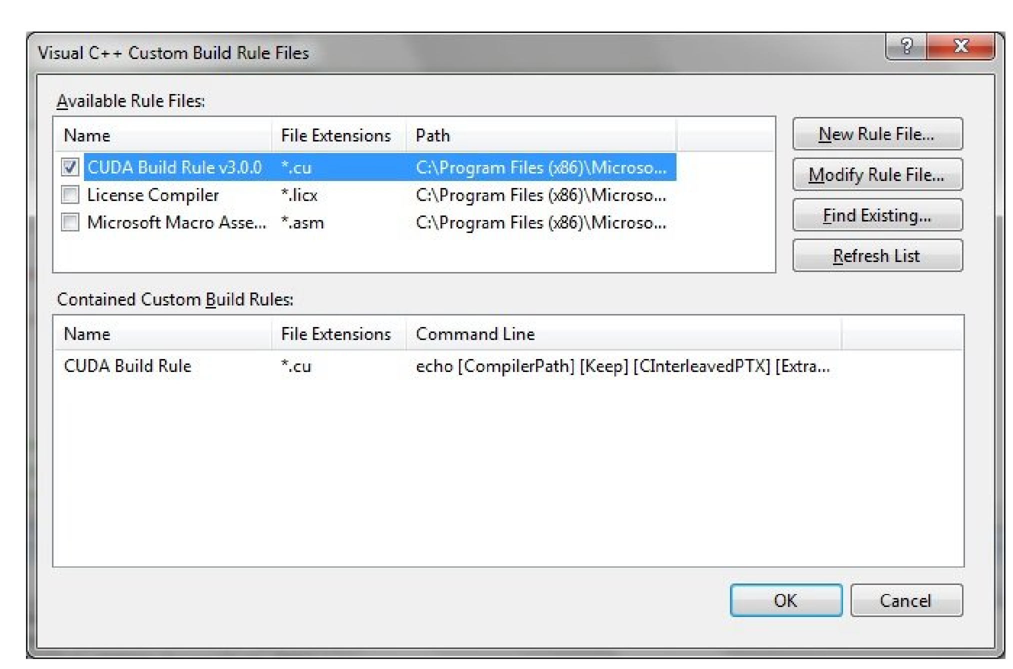
Méthode 3 – Complication de la vie

1. Ajoutez cuda.rules à votre projet.

    a. Cliquez avec le bouton droit sur votre projet > Custom Build Rules.

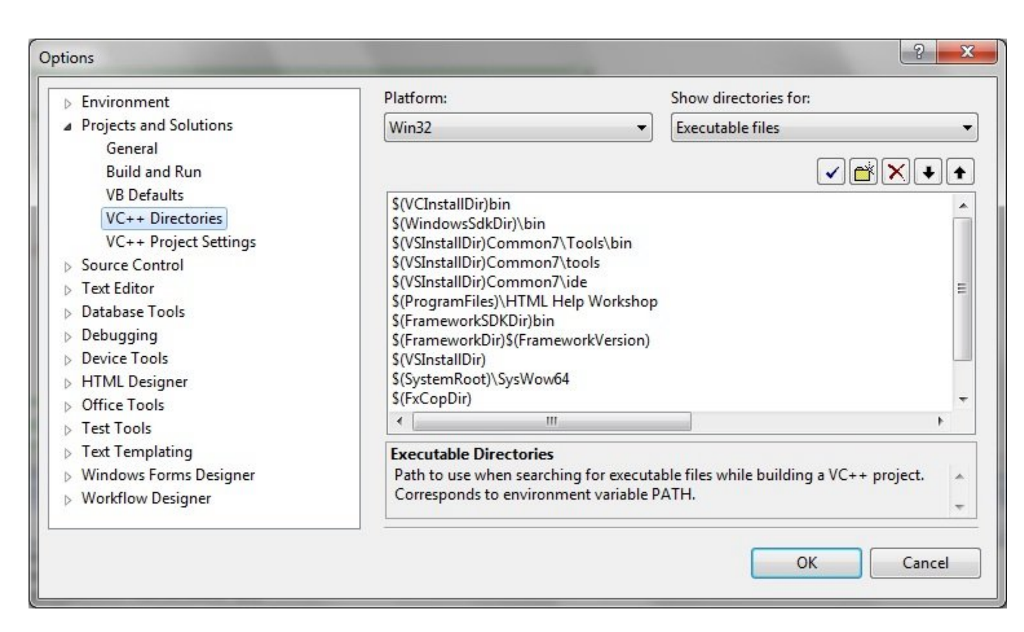


    b. Cliquez sur **New Rule File** et accédez à votre **% CUDA Folder%cuda_build_rule/cuda.rules**

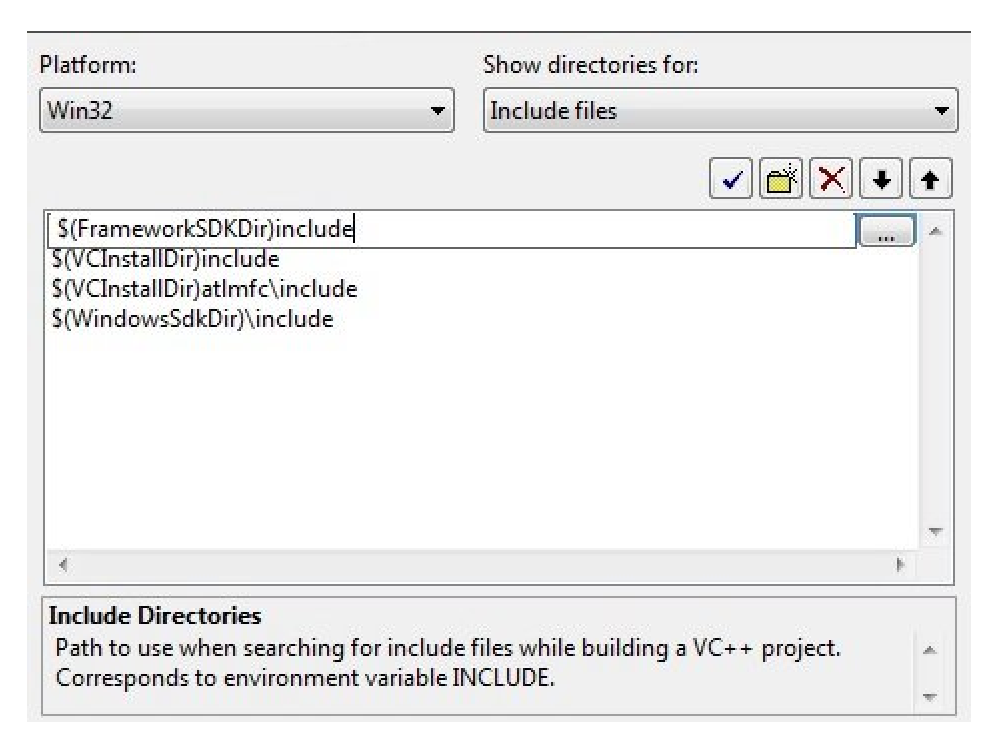


2. Configurez vos répertoires BIN, INCLUDE et LIB.

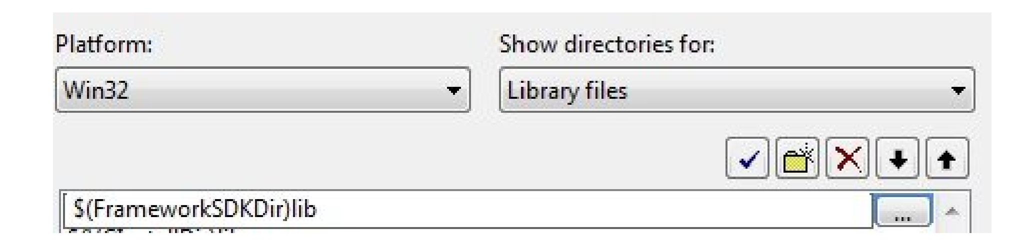
   a. Allez dans Tools > Options – > Projects and Solutions > VC ++ Directories



   b. Tout d'abord, spécifiez les répertoires **INCLUDE**. Sélectionnez **Include Files** dans le menu déroulant **Show Directories for**. Créez une nouvelle ligne en cliquant sur le bouton nouveau fichier et spécifiez **% CUDA % Directory Include**.

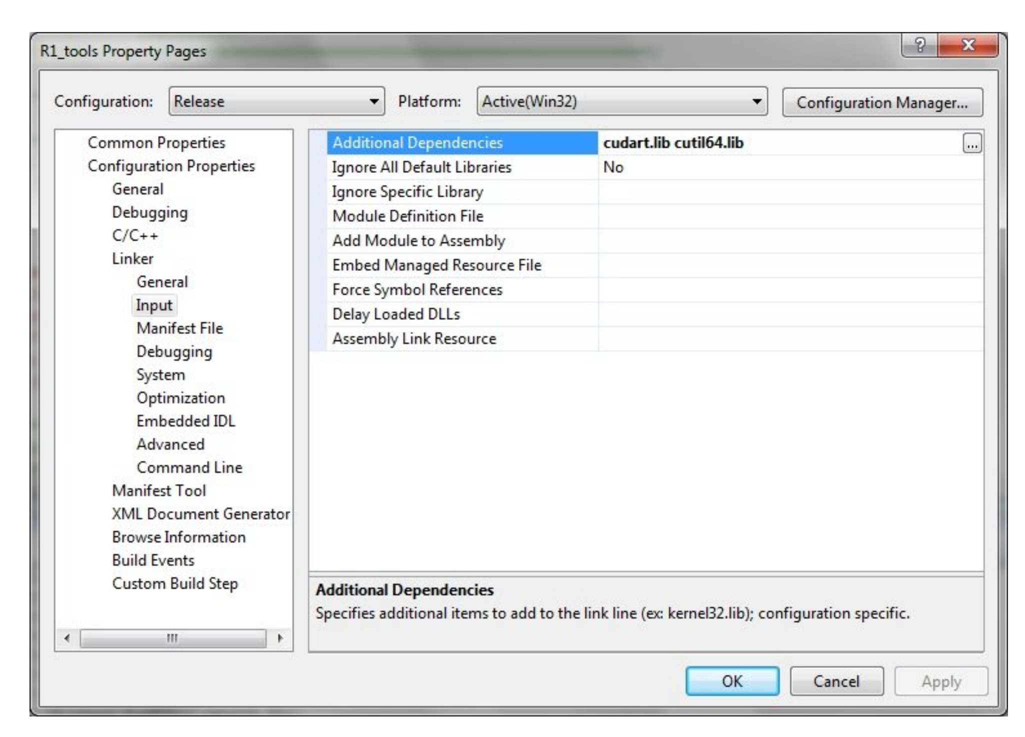


   c. De même, sélectionnez les **Library Files** dans la liste déroulante et spécifiez **% CUDA % lib**



3. Spécifiez le Runtime CUDA et les éditeurs de liens CUTILxxD.

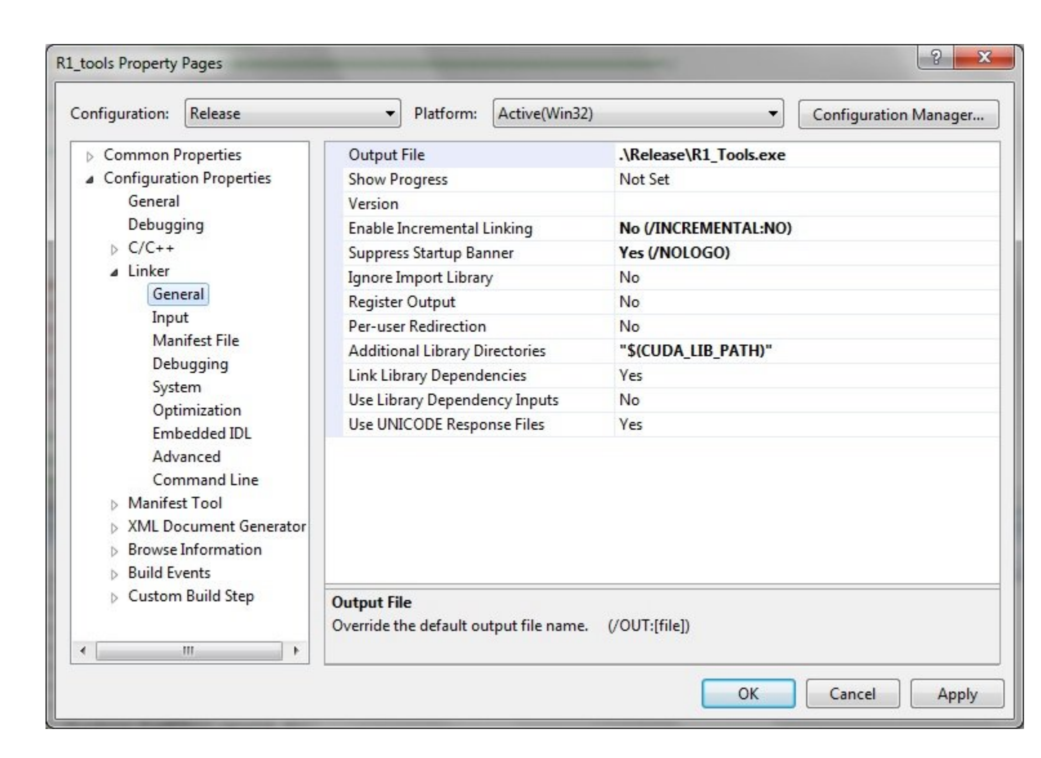
   a. Allez dans **Project > Properties > Configuration Properties > Linker > Input** et spécifiez deux fichiers dans le domaine des **Additionnal Dependencies** : cudart.lib et cutil32.lib (cutil64.lib si windows 64 bit).



Remarque

En mode Debug, il faut spécifier une autre librairie, spécifiquement conçue pour ce mode : CUTIL32D.lib ou CUTIL64D.lib.

   b. Allez ensuite à **Project > Properties > Configuration Properties > Linker > General et collez cette ligne dans le champ Additional Library Directories : $(CUDA_LIB_PATH).**



Et voilà, c'est fini. Vous pouvez maintenant essayer de **Rebuild** votre solution.

Récupérer la coloration syntaxique

1. Allez jusqu'au dossier "Microsoft Visual Studio 9.0\Common7\IDE".

2. Ouvrez le fichier utilisateur "usertype.dat" situé dans le dossier. Si le fichier n'existe pas, créez-en un portant ce nom.

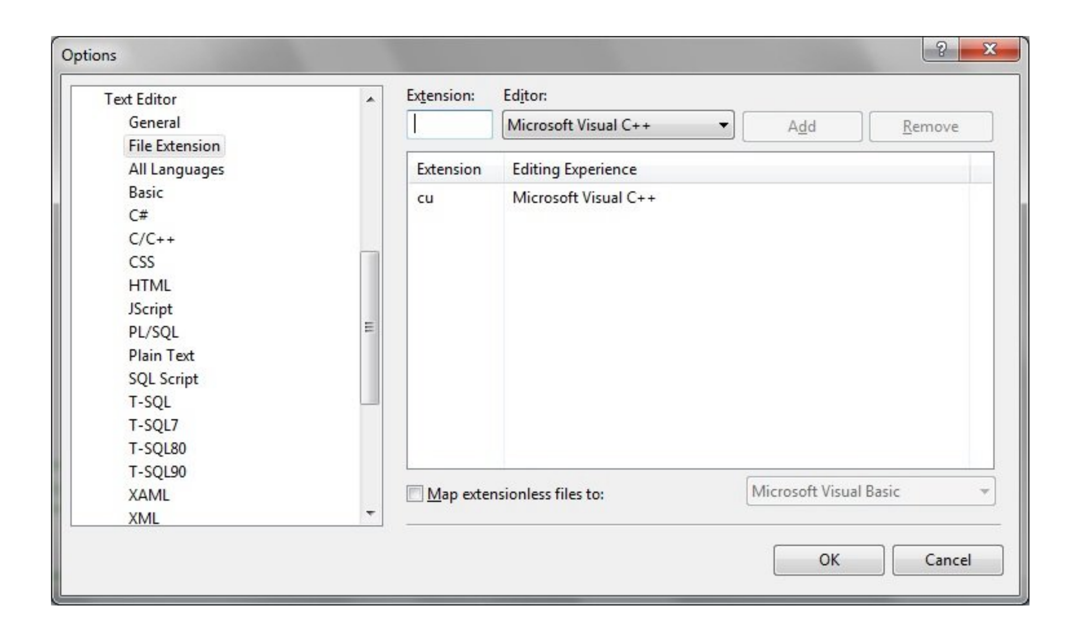
3. Ouvrez-le "%Program Files%\NVIDIA Corporation\NVIDIA CUDA SDK\C\doc\syntax_highlighting\visual_studio_8"

4. Collez le contenu de "usertype.dat" dans le "usertype.dat" précédemment ouvert/ créé.

5. Sauvegardez le fichier.

6. Lancez votre IDE et ouvrez Tools > Options

7. Sous Text Editor > File Extension, spécifiez l'extension "cu" comme un nouveau type (comme montré dans la figure ci-dessous).



Activer Intellisense pour CUDA

Activer la coloration syntaxique ne va pas activer Intellisense, l'auto-complétion, et surtout les aides à la navigation dans le code (les petits menus déroulants en haut de l'éditeur qui permettent d'aller directement à une fonction). Pour activer tout ça, il faut aller dans l'éditeur de registre, dans le dossier: :

HKEY_CURRENT_USER\Software\Microsoft\VisualStudio\9.0\Languages\Language Services\C/C++

Là, modifiez la clé NCB Default C/C++ Extensions et ajoutez-y les extensions .cu et .cuh, et le tour est joué !

